PA0 Flappy Bird

CIIC 3015

DUE: Feb 9, 2026 11:59 PM

Introduction

Hello everyone, welcome to the new semester! As you may know, you'll be doing a series of projects throughout your journey on this course. So let's get your computer ready to work with those! The focus of the Programming Assignment 0 (PA0 for short) is to go through the set-up process, and also display the general workflow of your future PAs. It is important that you succeed in this assignment, as it is the one that provides the foundation for all future work.

Setup

Download and Install Python 3.13

To set up Python 3.13, follow these steps:

Step 1: Download Python 3.13

1. Visit the official Python Downloads page.
2. Select **Python 3.13** and choose the appropriate installer for your operating system (Windows, macOS, or Linux).
3. Click **Download** and wait for the installer to finish.

Step 2: Install Python 3.13

1. Run the installer after downloading.
2. Make sure to check the box **"Add Python to PATH"** before proceeding.
3. Follow the installation steps and wait for it to complete.

Step 3: Verify Installation

1. Open a terminal (Command Prompt, PowerShell, or Terminal on macOS/Linux).
2. Type the following command and press Enter:

```
None
python --version

# OR

python3 --version
```

3. You should see Python 3.13.x displayed.

Let's Get PyCharm Ready! 🚀

YouTube Video: Install PyCharm IDE with Python 3.13 on Windows 10/11 [2024] || ...

To get started with PyCharm Community, the free and open-source edition of PyCharm, follow these steps:

Step 1: Download PyCharm Community

1. Visit the PyCharm Download Page.
2. Under the Community edition (which is free for everyone), click **Download**.
3. Choose the version for your operating system (Windows, macOS, or Linux).

Step 2: Install PyCharm

1. Once the download is complete, run the installer.
2. Follow the on-screen instructions to install PyCharm on your computer.
3. Open PyCharm after installation is finished.

Step 3: Get a Free JetBrains Education License

- If you're a student, you can get PyCharm Professional Edition for free with a JetBrains Education License!

1. Go to the JetBrains Student License page.
2. Click **Apply Now** and verify your student status using your university email (e.g., john.doe@upr.edu).
3. Once verified, you can use PyCharm Professional Edition for free while you're a student! 🚀

Setting Up the Python Interpreter in PyCharm

After installing Python 3.13, you need to set it as the interpreter in PyCharm.

Step 1: Open PyCharm and Create a New Project

1. Launch PyCharm.
2. Click **New Project**.
3. Under **Location**, set a directory for your project. We recommend that you create a new folder in your laptop to save all your CIIC 3015 PyCharm projects.

Step 2: Configure Python Interpreter

1. In the **New Project** window, under **Python Interpreter**, click **Add Interpreter**.
2. Select **System Interpreter**.
3. Browse to the location where Python 3.13 is installed (usually C:\\Python313\\python.exe on Windows or /usr/bin/python3.13 on macOS/Linux).
4. Click **OK** to set it as the default interpreter.

Step 3: Verify Interpreter Setup

1. Open **Terminal** in PyCharm (bottom panel).
2. Type:
   python --version
   It should show Python 3.13.x.

Now, you're all set to start coding in Python 3.13 using PyCharm! 🚀

You should set your preferred Python interpreter every time you create a new project. This is done on purpose because it gives you the option to use different versions of Python for each project. However, in this course we will always use the same version 3.13.

💥Git and GitHub - Version Control

Let's break down the instructions for setting up Git and GitHub:

▶ How To Use GitHub For Beginners

Step 1: Creating a FREE GitHub Account

1. Go to GitHub's website and sign up for an account.
2. Use your @upr.edu email to register.
3. Pick a meaningful GitHub username, like **“john.doe”** or **“john.doe12”**, that relates to your real name.

Step 2: Configuring Git Locally

1. Open **Terminal** (macOS/Linux) or **Command Prompt** (Windows).
2. Install git:
    - **macOS:** Run brew install git
    - **Windows:** Download Git and run the installer.

- Run the following commands to set your Git username and email (replace with your actual GitHub details):
  git config --global user.name "your GitHub username"

3. git config --global user.email "your GitHub email"
   NOTE: You won’t see any confirmation or output. This is normal.
4. To ensure everything is set up correctly, run:
   git config --list
   This will display your Git configuration—check that your username and email are correct.

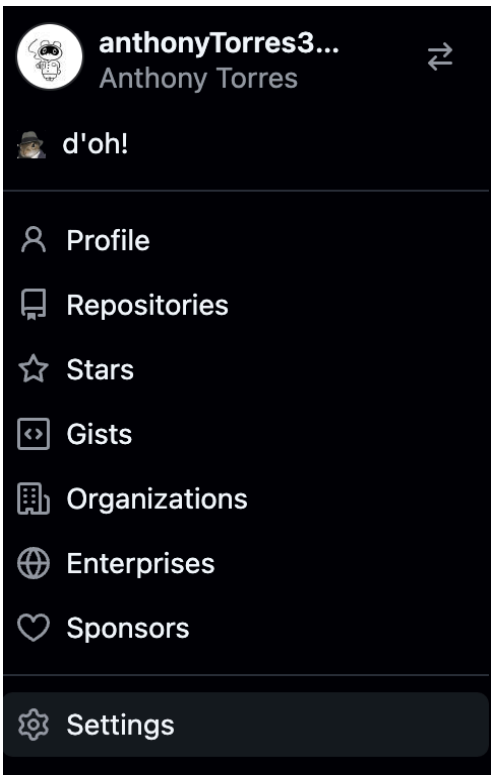
🤝 Project: Accepting and Cloning PA0

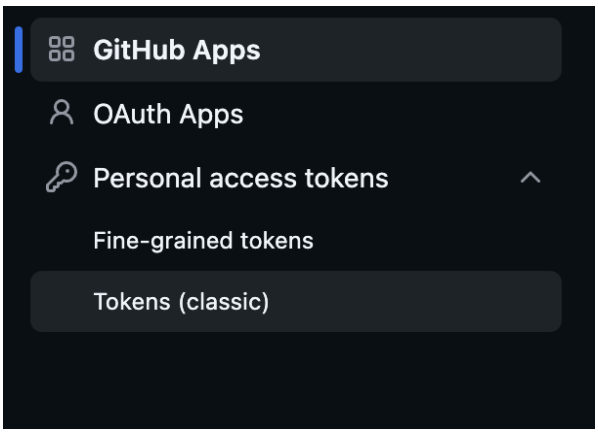
With Git and GitHub in place, it’s time to accept and clone your very first programming assignment—PA0! Let’s get your project onto your machine:

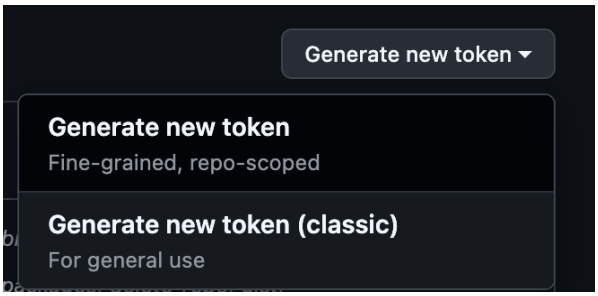
Cloning the Assignment Repository

To clone the repository, you'll need a **GitHub token** for authentication.

1. Go to **GitHub's Settings → Developer Settings → Personal Access Tokens → Tokens (classic) → Generate new token**.







2. Select the appropriate scopes for your token (such as repo) and generate the token.
3. **Save the token**—you'll need it for authentication during cloning and GitHub will not show it to you again.

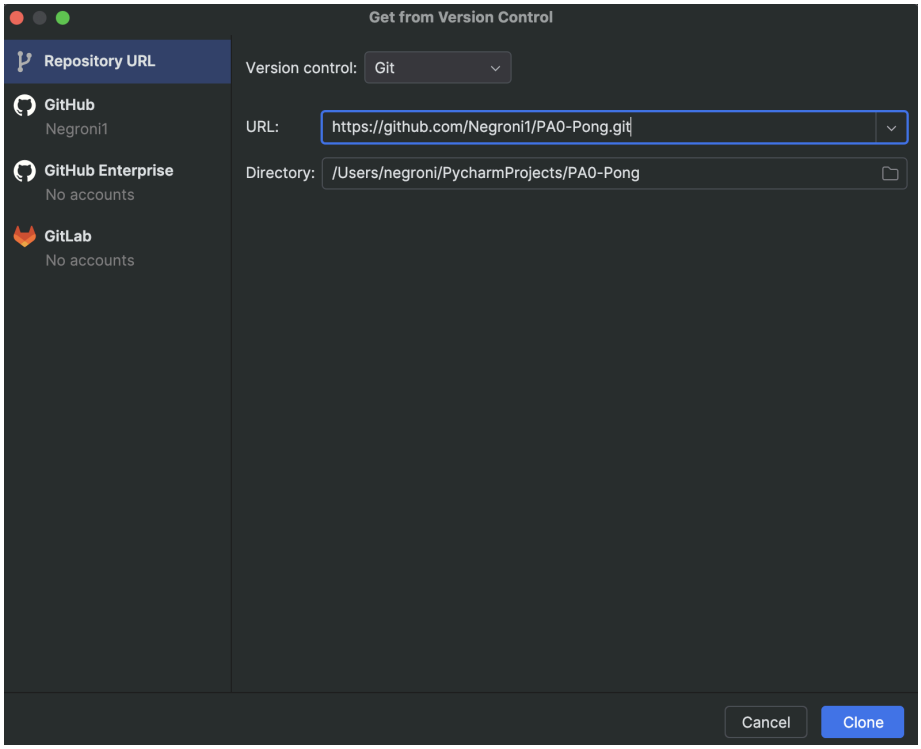
Accept the Assignment:

1. Click here OR copy and paste the following link to accept or join PA0 https://classroom.github.com/a/ttF8LaKz. One of the partners must create the group and the other partner must join it. DO NOT WORK ON THE SAME LAPTOP.
2. After accepting, GitHub will create a new repository for your assignment and display a link with your team name similar to the following

(https://github.com/UPRM-CIIC315-F24/pa0-yourteamname).

Clone the Repo in PyCharm:

1. Open PyCharm and navigate to **VCS → Get from Version Control** or **File → Project from Version Control**.



2. Select **Git** as your Version Control type and paste your clone URL.
3. When prompted for authentication, use your GitHub token instead of your password.
4. Choose your local directory and hit **Clone**.
5. PyCharm will open your project—time to dive in and code!

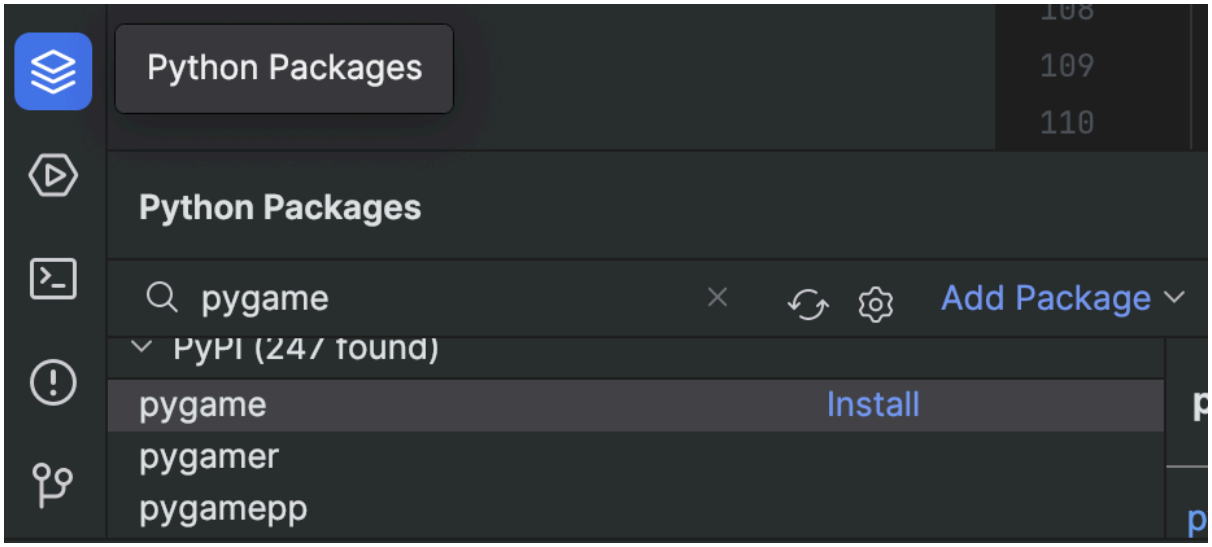
Now you can start working on the project! 🎉

🎮 Pygame - Let’s Start Gaming!

YouTube Video: The ultimate introduction to Pygame

Now to be able to run the project we need the Python module Pygame!

In your open project in PyCharm go to Python Packages directly on the left bar. In the Python Packages tool window, search for pygame and then click the Install button.

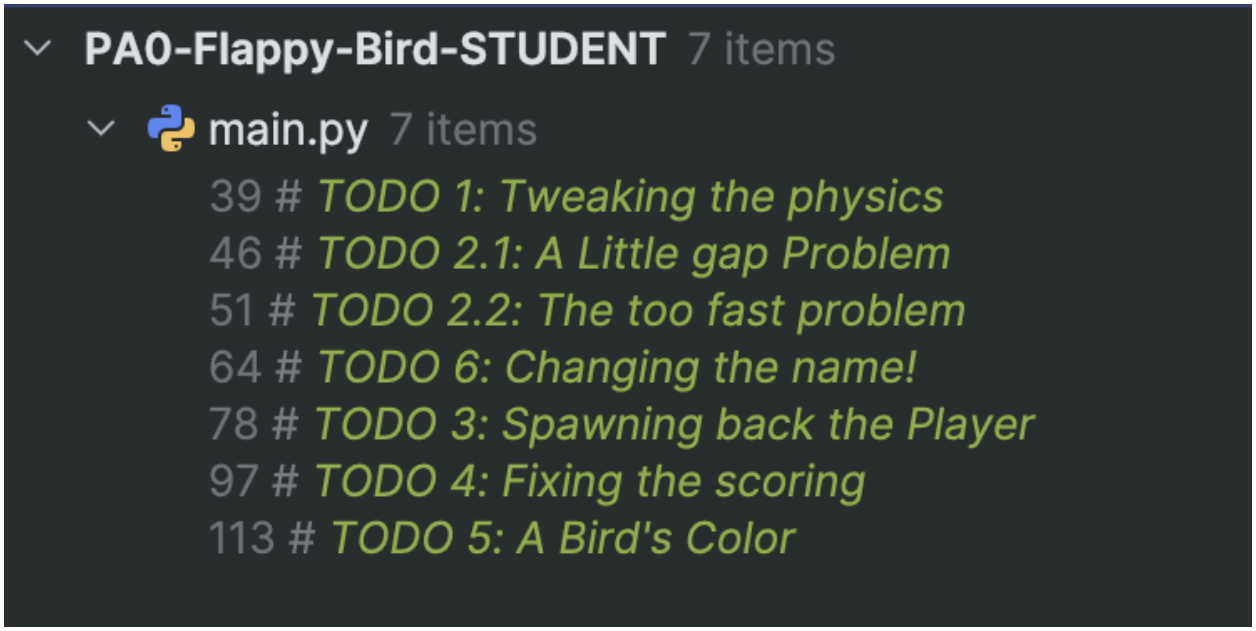


Congrats! Now you are ready to start coding!

To run the game open the main.py Python file and click the *play* button at the top right. To play the game, press the *spacebar* to start the player movement.

📝 Tasks: Let’s Tackle Those To-Do’s!

To keep track of what needs to be done, check out the TODO section in the left menu or look for TODO comments in the code.


PA0-Flappy-Bird-STUDENT 7 items
main.py 7 items
39 # TODO 1: Tweaking the physics
46 # TODO 2.1: A Little gap Problem
51 # TODO 2.2: The too fast problem
64 # TODO 6: Changing the name!
78 # TODO 3: Spawning back the Player
97 # TODO 4: Fixing the scoring
113 # TODO 5: A Bird's Color

How to submit each tasks

To submit the task on your GitHub repository, follow the next steps:

Commit Your Changes

Stage Your Changes

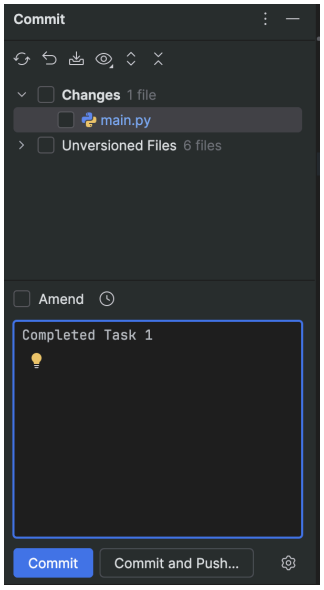
Before you can commit your changes, you need to stage them. This tells Git which changes you want to include in the next commit. In PyCharm, go to the Git menu and select Commit. A dialog will appear showing all the changes you’ve made. Check the boxes next to the files you want to include to stage them. For example:

Write a Commit Message

In the same dialog, you'll see a text box where you can write a commit message. This message should briefly describe what changes you made (e.g., "Completed Task 1"). Good commit messages make it easier to understand the history of your project.

Commit the Changes

After writing your commit message, click the Commit button. This records the changes in your local Git repository only. At the end it should look something like this:



Push Your Changes to GitHub

Push to GitHub

Once you've committed your changes locally, you need to push them to your remote GitHub repository to make them available to your partner online. Go to the Git menu, and then choose Push. A dialog will appear, showing the commits that will be pushed. Click the Push button to upload your changes to GitHub!

Let's Start Tackling Those To-Do's!

Task 1: Tweaking the Physics 🍎

(Suggestion: Teammate A)

When starting to play you might notice that the bird immediately goes down not giving you a chance to flap upwards. This is a problem with the gravity of the player. Start tweaking the gravity so that it's in a state where the game is playable. In addition to this you will have to add a little something to the **bird_velocity** variable.

> 💡 **Important** You don't have to insert a specific number for the gravity to receive the points; it has to be a number where the game is playable.

Task 2: The Pipe Problem 🔧

Task 2.1: A little gap problem 🟢

(Suggestion: Teammate B) d

The main thing to a flappy bird is passing through the gap between pipes but for some reason the gaps are a little small don't you think? Find what variable is in charge of adjusting the height of the gap and tweak it so it's big enough for the bird to pass through!

Task 2.2: The too *fast* problem 🚀

(Suggestion: Teammate A)

Something else you probably noticed is how fast the pipes are moving. Find the variable in charge of this and tweak it to your liking!

Task 3: Player Spawn ⭐

(Suggestion: Teammate B)

After crashing and losing you might notice that when you try to reset the player doesn't appear in the player screen.

> 💡 **Hint** What variable stores the y coordinates of the bird

Task 4: Fixing the scoring 🏆

(Suggestion: Teammate A)

In the if statement that checks whether the player passed through the pipes you need to ensure the score increases by 1 when the player passes through the pipe.

💡 **Hint**	The logic for checking whether the player passed through the pipes already exists maybe start looking in that area to see what's causing it not to increase

Task 5: Change the player Color 🎨

(Suggestion: Teammate B)

Head to the variable named PLAYER. The ball is currently (255, 255, 255) (which is white). Get creative and change it to any color you fancy, for example: red.
To find the colors available go to https://www.pygame.org/docs/ref/color_list.html
Documentation on how to draw the rect:
https://www.pygame.org/docs/ref/draw.html#pygame.draw.rect

Task 6: Change name 🐞

(Must Be Done by Teammate A and B)

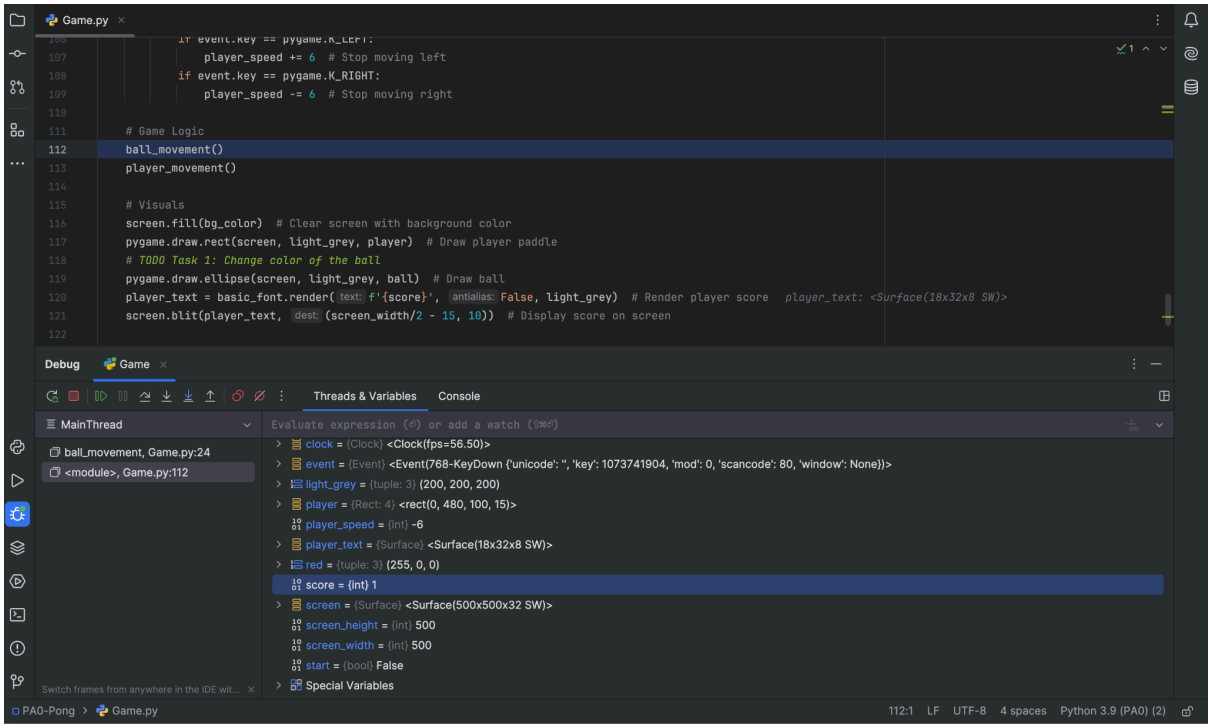
The PyCharm debugger is a powerful tool that helps you inspect and troubleshoot your code by allowing you to pause execution, step through code, and examine the state of your program at specific points. Here's a breakdown of how to use it effectively:

Setting Breakpoints

What are Breakpoints? Breakpoints are markers you place in your code to tell the debugger to pause execution at that specific line. Now to wet a breakpoint you need to click in the line number next to the line where you want to pause. A red dot will appear, indicating a breakpoint.

Starting the Debugger

Once you've set your breakpoints, you can start the debugger by clicking the green bug icon in the top right corner of PyCharm (or right-click the script and choose "Debug"). PyCharm will start executing your program. When it hits a breakpoint, it will pause execution and open the Debug tool window at the bottom



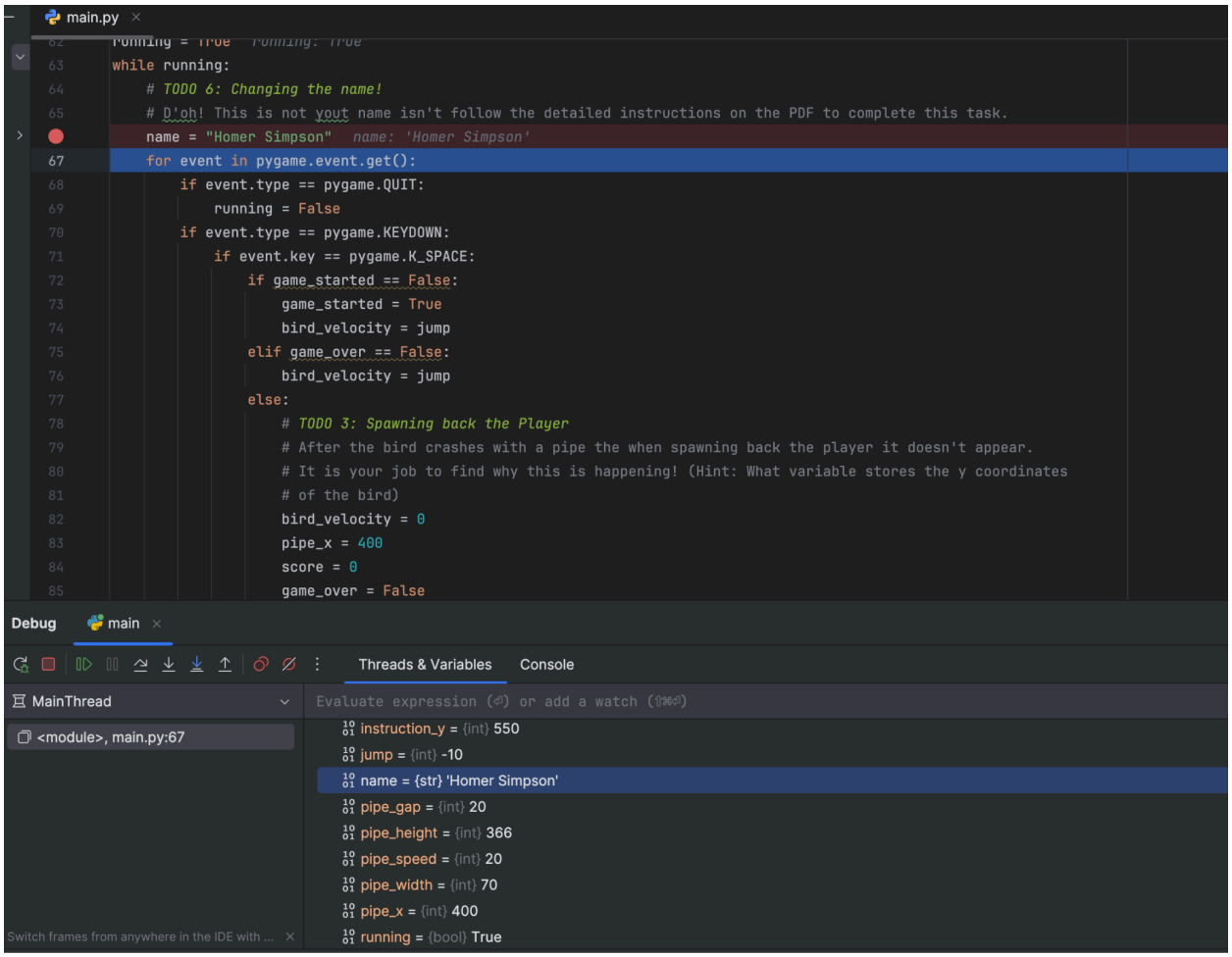
Navigating Code with the Debugger

The more important navigation actions that you need to know are:

Icon	Name	Function
	Step Over	Executes the current line of code and moves to the next one. If the current line is a function call, the debugger will execute the entire function without stepping into it.
	Step Into	If the current line contains a function call, the debugger will go inside that function and pause at the first line.
	Step Out	If you are inside a function, this command will run the rest of the function and pause after returning to the caller.
	Resume Program	Continues running the program until the next breakpoint is hit or the program ends.

To complete this task:

- Go to TODO Task 6: Add your name
- Change the variable name to **your name** (e.g. Juan del Pueblo)
- Place a breakpoint and run the debugger
- Take a screenshot of the debugger running and the value of name change in the debug pane and ***upload it in the assignment form in Moodle***. An example of the screenshot:


main.py
while running:
# TODO 6: Changing the name!
# D'oh! This is not yout name isn't follow the detailed instructions on the PDF to complete this task.
name = "Homer Simpson"
for event in pygame.event.get():
if event.type == pygame.QUIT:
running = False
if event.type == pygame.KEYDOWN:
if event.key == pygame.K_SPACE:
if game_started == False:
game_started = True
bird_velocity = jump
elif game_over == False:
bird_velocity = jump
else:
# TODO 3: Spawning back the Player
# After the bird crashes with a pipe the when spawning back the player it doesn't appear.
# It is your job to find why this is happening! (Hint: What variable stores the y coordinates
# of the bird)
bird_velocity = 0
pipe_x = 400
score = 0
game_over = False
Debug
main
Threads & Variables
Console
MainThread
<module>, main.py:67
Evaluate expression (⏎) or add a watch (⇧⌘⏎)
instruction_y = {int} 550
jump = {int} -10
name = {str} 'Homer Simpson'
pipe_gap = {int} 20
pipe_height = {int} 366
pipe_speed = {int} 20
pipe_width = {int} 70
pipe_x = {int} 400
running = {bool} True

Each team member must upload the picture into his/her own evaluation form with **their** name.

Task 7: Adding Noise! 🔊

(Suggestion: Teammate A)

Download or create a .wav sound file for when the ball hits a paddle. Implement sound playback using the pygame.mixer library

Ensure the sound plays when the player scores a point. Remember to also upload the .wav file to Github!

Task 8: Causing a Conflict 🔥

(Must be done by Teammate A and then B)

One of the things that you will possibly be encountering when working on a project with someone else is a merge conflict.

💡 **What is a Merge Conflict**	*A merge conflict happens when two branches change the same part of a file. Git can't decide which change to keep, so you have to choose. You must resolve the conflict before you can finish the merge.* **Source***:* Git Merge Conflicts

💡**Why do merge conflicts happen?**	*Merge conflicts usually happen when you merge branches that changed the same lines in a file. This is common in collaborative projects or when working on long-lived branches.* **Source**: Git Merge Conflicts

Teammate A:

- Change the bird_velocity variable (e.g., from 8 to 14).
- Commit and **push** the change.

Teammate B:

- WAIT for teammate A to push their changes BUT **DO NOT PULL**, also change bird_velocity (from 7 to 10).
- Commit and push (**this will create a merge conflict!**).

Step-by-Step Conflict Resolution:

YoutTube Video: Resolving Git Conflicts: The Easy Way

Teammate B pulls from GitHub (git pull).

1. Git will show a conflict in game.py.Open game.py. You'll see conflict markers:

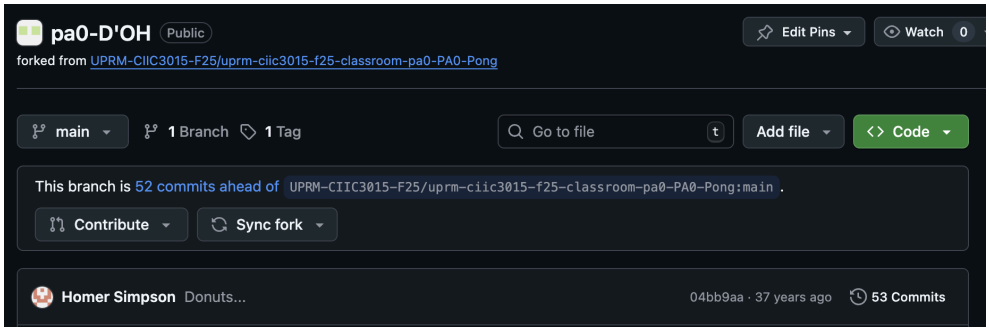
```
None
<<<<<<< HEAD

bird_velocity = 7
=======
bird_velocity = 10
>>>>>>> teammateA-branch
```

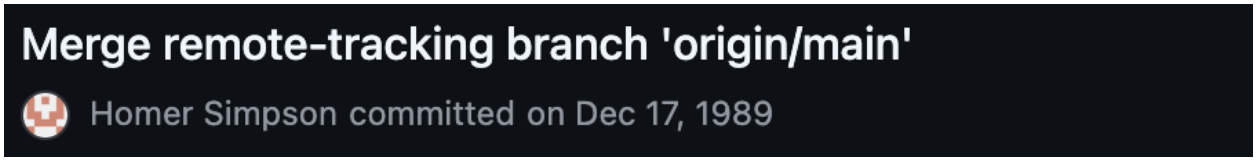
2. Decide on the correct value (e.g., ball_speed_x = 10).
3. Remove conflict markers (<<<<<<<, =======, >>>>>>>).
4. Save and commit the resolved file.
5. Push the changes back to GitHub.

Merge conflict solved! 🎉

How to know if you did it?

Go to where it says x amount of commits + commit when you go there search if you see a commit that looks something like this





Task 9: ReadME 📝

You are almost finishing this project. Congratulations for getting this far! You probably already noticed that there is another file aside of main.py. It's the readme file! Here you are going to fill out the readme file with your name, student number, email address, and list bonus that you have done!

🎯 Final Submission

To submit your project each team member must submit an evaluation form independently under PA0 in Moodle. One of the fields in the evaluation form will ask for a GitHub tag for your code. To generate a tag follow the following steps:

- Go to your repo and click on the Releases link usually found under the repository name.
- Click on the Draft a new release button.
- Under the Tag version field, enter the name of the tag you want to create (e.g v1.0.0).
- Fill in the release title and description (optional).
- Click on Publish release to create the tag and release.
- Enter the name of the tag in the evaluation form.

Congratulations on completing your first project assignment!

If you're aiming for bonus points, feel free to enhance your game by adding new features and making it truly exceptional. **Bonus** points range from 1 to 15, based on the TA's discretion. For any questions, refer to the rubric, ask in the PA0 Discord channel, or stop by during office hours! Happy coding! 💻

Bonus suggestions

- Keep a high score
- Multiplayer functionality
- Add music
- Level difficulties
- Add images